REpresentational State Transfer (REST)

Software Languages Team
University of Koblenz-Landau
Ralf Lämmel and Andrei Varanovich

REST -- ***a set of principles*** that define how Web standards, such as HTTP and URIs, are supposed to be used in Web applications.

Resources and Resource Identifiers

- The key abstraction of information in REST is a ***resource.***
- Each resource has a ***resource identifier.***

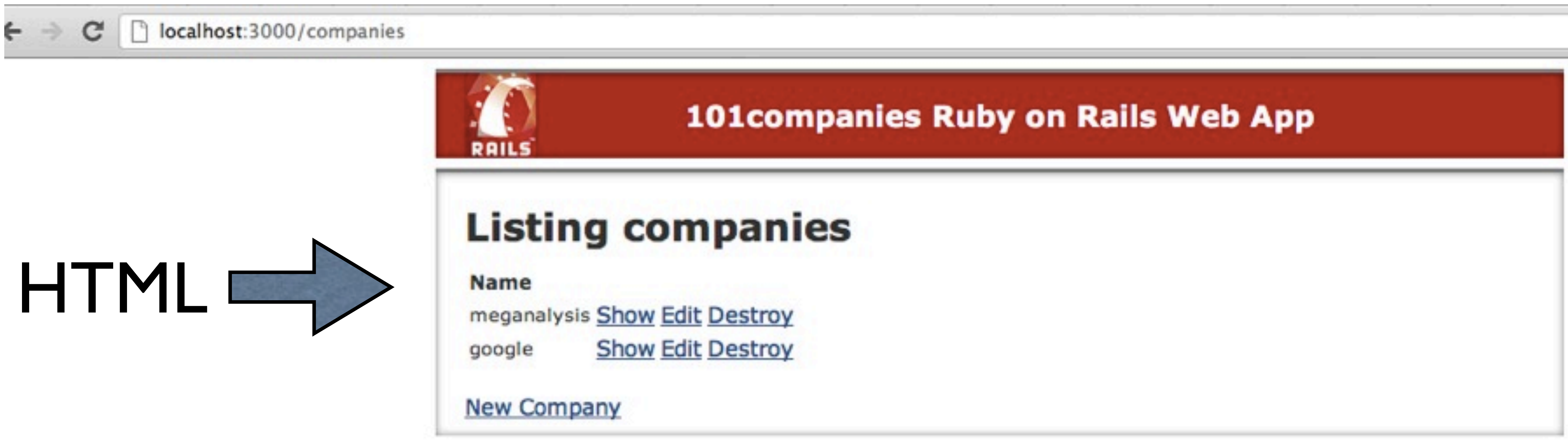
Examples of identifiers

- http://example.com/customers/1234
- http://example.com/orders/2007/10/776654
- http://example.com/products/4554
- http://example.com/processes/salary-increase-234

Resources can have multiple representations,
e.g., JSON/XML/HTML.

Example: the resource of 'all companies'

HTML

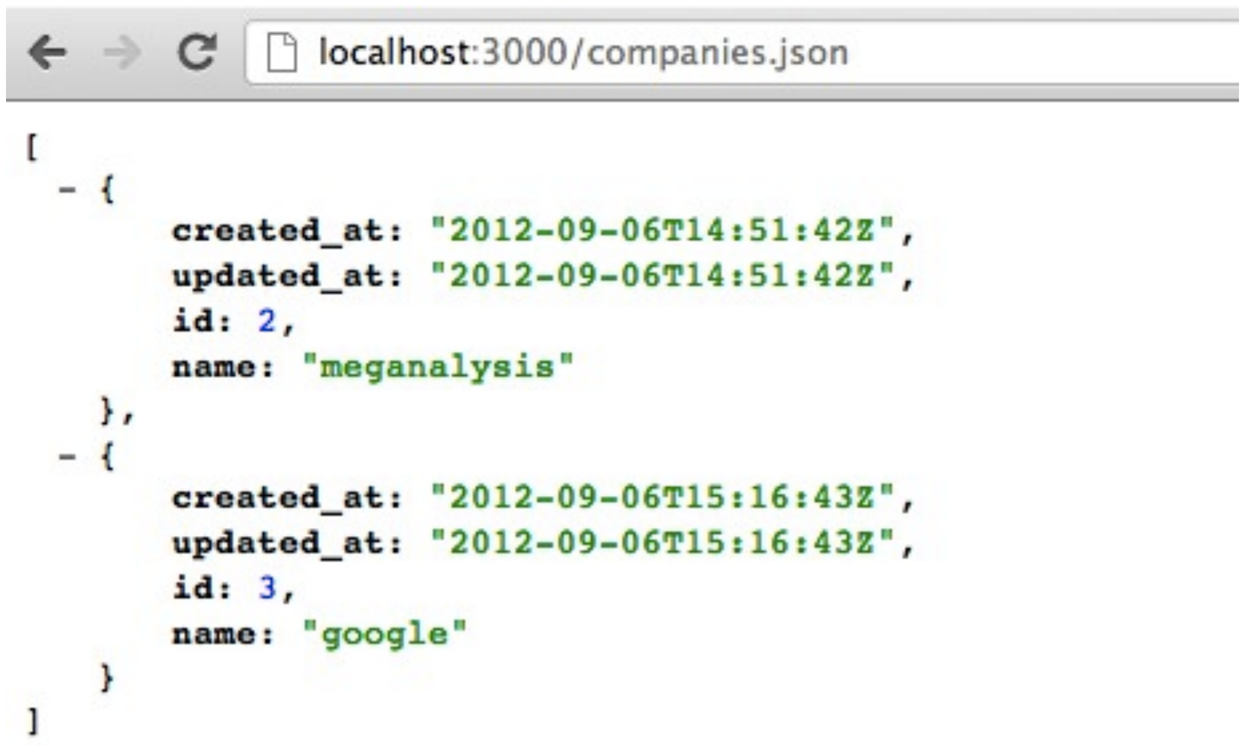


101companies Ruby on Rails Web App

Listing companies

Name			
meganalysis	Show	Edit	Destroy
google	Show	Edit	Destroy

New Company



```
[
  - {
        created_at: "2012-09-06T14:51:42Z",
        updated_at: "2012-09-06T14:51:42Z",
        id: 2,
        name: "meganalysis"
    },
  - {
        created_at: "2012-09-06T15:16:43Z",
        updated_at: "2012-09-06T15:16:43Z",
        id: 3,
        name: "google"
    }
]
```

JSON

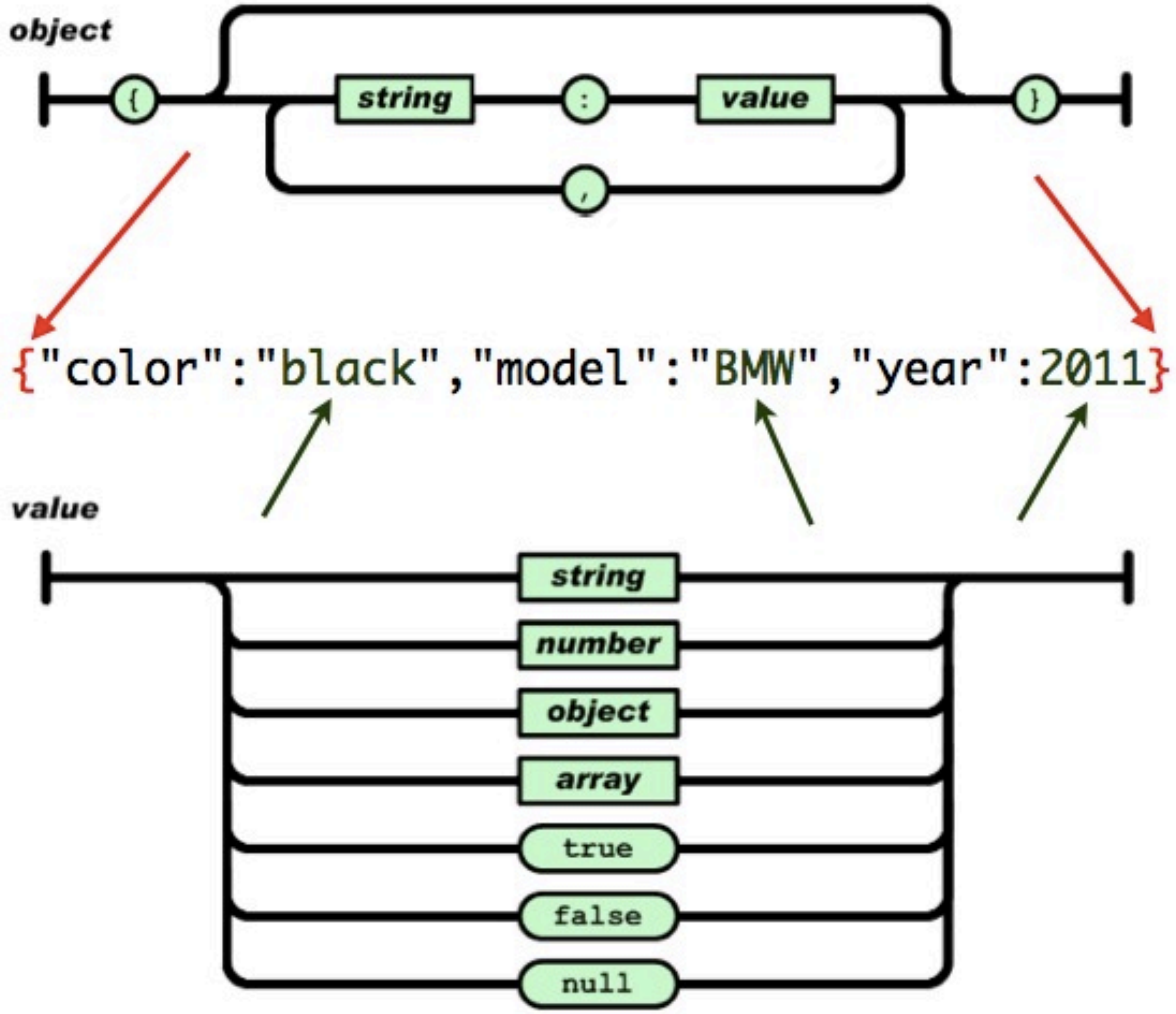
What is JSON?

JSON (JavaScript Object Notation)

- It is easy for humans to read and write.
- It is easy for machines to parse and generate.
- Native support in JavaScript

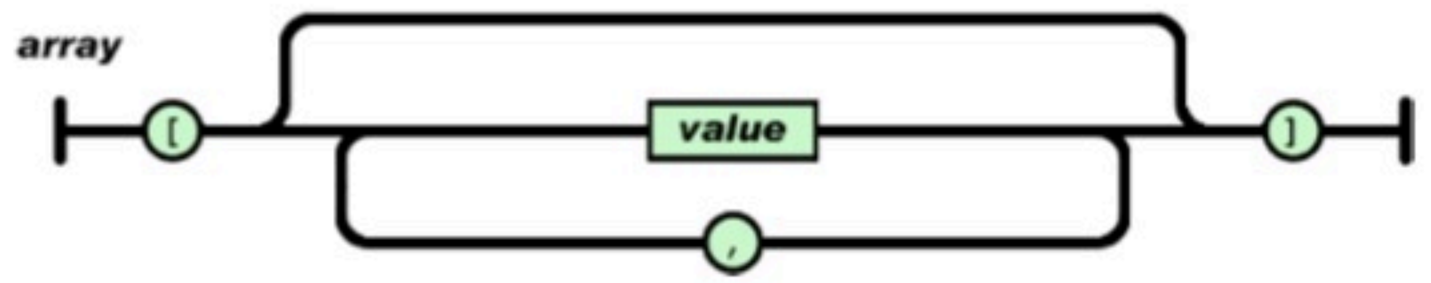
http://www.json.org/

JSON



```
{"color":"black","model":"BMW","year":2011}
```

Arrays in JSON



```
[ {"color":"black","model":"BMW","year":2011},
   {"color":"white","model":"Audi","year":2010},
{“x”:5} ]
```

JSON in Java

```java
public class Car {
   private String color;
   private String model;
   private Integer year;

   public Car(String color, String model, Integer year){
      this.color = color;
      this.model = model;
      this.year  = year;
   }
   .....
}
```

```java
Gson gson = new Gson();
Car car = new Car("black", "BMW", 2011);
String json = gson.toJson(car);
```

```
{"color":"black","model":"BMW","year":2011}
```

Back to REST

Remember: Hypertext Transfer Protocol

http://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol

• **GET**	Request representation for resource
• **HEAD**	Like GET but without response body
• PUT	Upload representation for resource
• **POST**	Submit data for resource
• OPTIONS	Query for available methods
• CONNECT	Facilitate SSL-encrypted communication
• DELETE	Delete specified resource
• TRACE	Return request as it arrived at server
• PATCH	Partial modification of resource

RESTful Web Service HTTP methods

- ***Collection*** URI, such as http://example.com/companies/
- GET: **List** the URIs and perhaps other details of the collection's members
- PUT: **Replace** the entire collection with another collection.
- POST: **Create** a new entry in the collection. The new entry's URL is assigned automatically and is usually returned by the operation.
- DELETE: **Delete** the entire collection.

RESTful Web Service HTTP methods

- ***Element*** URI, such as http://example.com/companies/32
- GET: **Retrieve** a representation of the addressed member of the collection, expressed in an appropriate Internet media type.
- PUT: **Replace** the addressed member of the collection, or if it doesn't exist, **create** it.
- POST: Treat the addressed member as a collection in its own right and **create** a new entry in it.
- DELETE: **Delete** the addressed member of the collection.

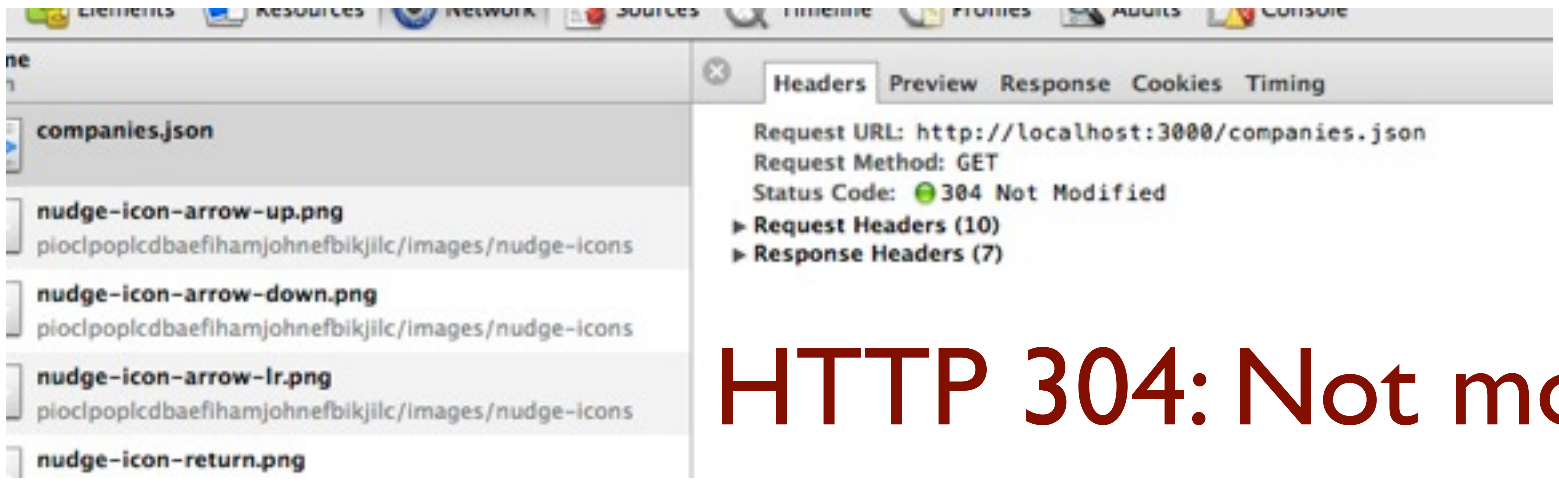
DEMO

101implementation: rubyonrails

- Show the look and feel of the implementation
- Explain the URL scheme in relation to REST
- Show JSON vs. HTML representation

Properties of REST

- Client-server
- Stateless: user data is not stored between requests
- Cache



companies.json
nudge-icon-arrow-up.png
pioclpoplcdbaefihamjohnefbikjilc/images/nudge-icons
nudge-icon-arrow-down.png
pioclpoplcdbaefihamjohnefbikjilc/images/nudge-icons
nudge-icon-arrow-lr.png
pioclpoplcdbaefihamjohnefbikjilc/images/nudge-icons
nudge-icon-return.png

```
Request URL: http://localhost:3000/companies.json
Request Method: GET
Status Code: 304 Not Modified
▸ Request Headers (10)
▸ Response Headers (7)
```

HTTP 304: Not modified

Summary

You learned about ...

- the REST architecture pattern,
- working with "resources" via HTTP,
- the JSON format,
- and some bits of Ruby on Rails.

Resources

- A Brief Introduction to REST: http://www.infoq.com/articles/rest-introduction
- Architectural Styles and the Design of Network-based Software Architectures: http://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm